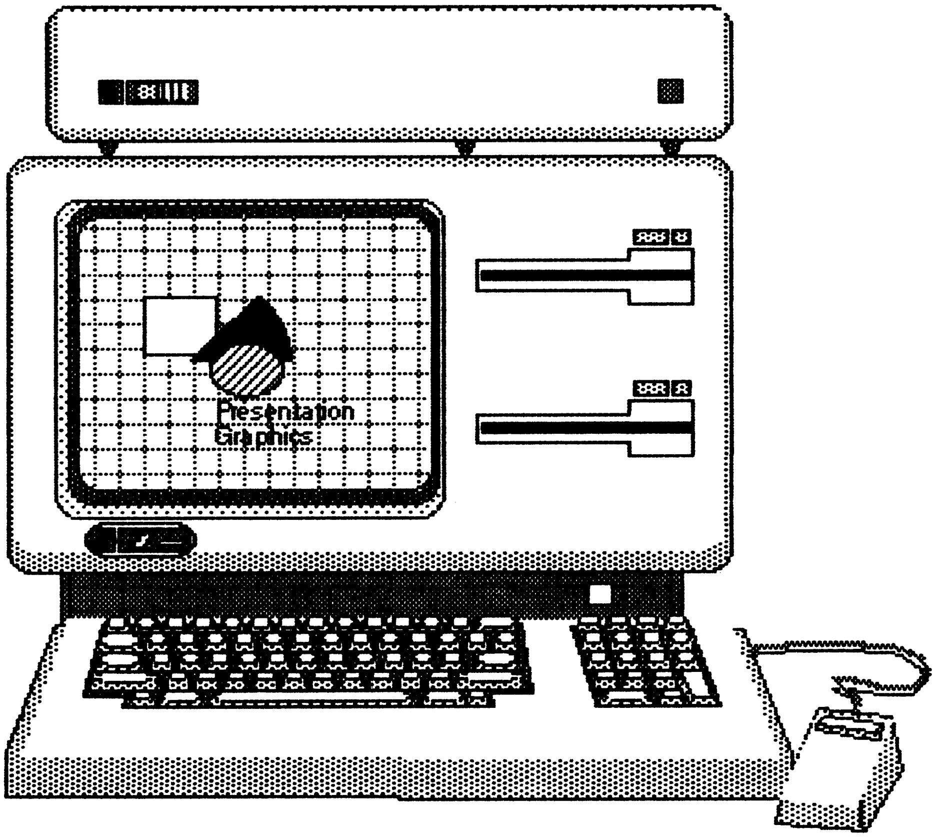
Apple Lisa Personal Computer
1983 to 1985

History of LisaDraw



David T. Craig - 736 Edgewater, Wichita, Kansas 67230 - (316) 733-0914

Racing How Apple Gets Its Software Out the Door

a and Macintosh are Colorful

By Brian Cutter

Have you ever wondered how a piece of software gets from idea to finished product? Who gets product ideas? Who develops them? Apple has a rather, shall we say, unique product development cycle. Since *MacDraw*, Apple's new graphics program, is nearing the end of its development cycle, let's use it as an example to illustrate how Apple develops and markets products.

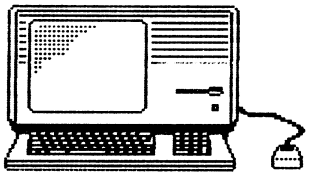
MacDraw is a powerful graphics editor that allows you to compose all kinds of drawings such as diagrams, flow charts, technical drawings, maps, blueprints, and organizational charts, from a palette of circles, boxes, polygons, lines, and text. *MacDraw* is based on *LisaDraw* and uses a similar palette of shapes that can be selected, enlarged, reshaped, and moved. Like *LisaDraw*, *MacDraw* is object-oriented; this means that a composition is represented as a collection of individual objects (shapes, lines, and text) that may be edited in groups or separately.

The story of *MacDraw*'s development actually begins more than three years ago with the development of *LisaDraw*, which is very similar to, and in fact a forerunner of, *MacDraw*. We can explore Apple's product development process by taking a look at the development of both programs: *MacDraw* and *LisaDraw*. Let's start with the beginning of the thirty-two-bit family, the Lisa.

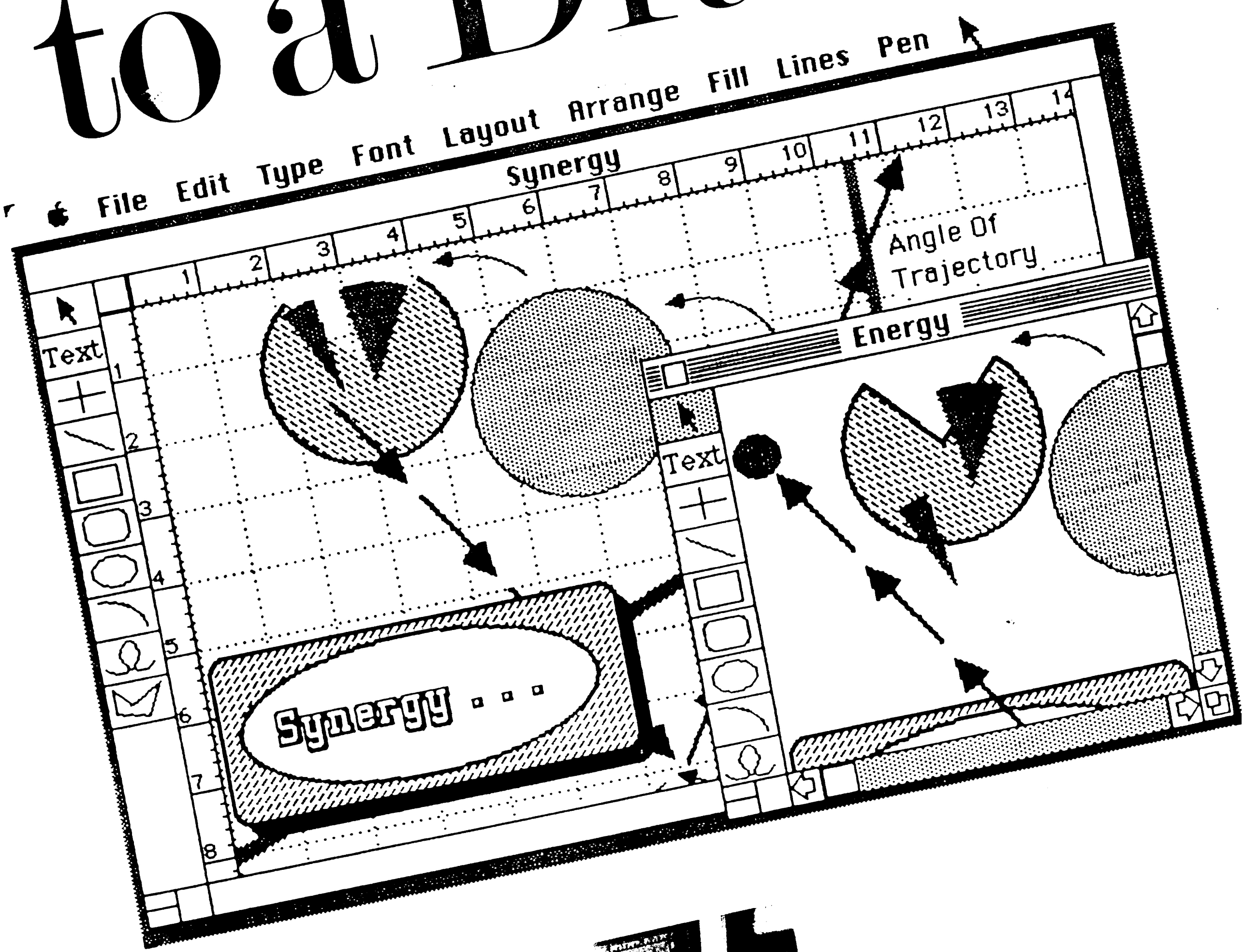
When you want to talk about the marketing of Lisa products, a good person to see is Randy Battat. As product manager for Lisa and Lisa products, Battat is in charge of setting Lisa's wholesale price, monitoring production levels, and, of course, overseeing the development of Lisa products.

Brian Cutter is a technical writer at Dysan Corporation in the company's Software Publishing Division, as well as a free-lance writer and photographer. His brother is Mark Cutter.

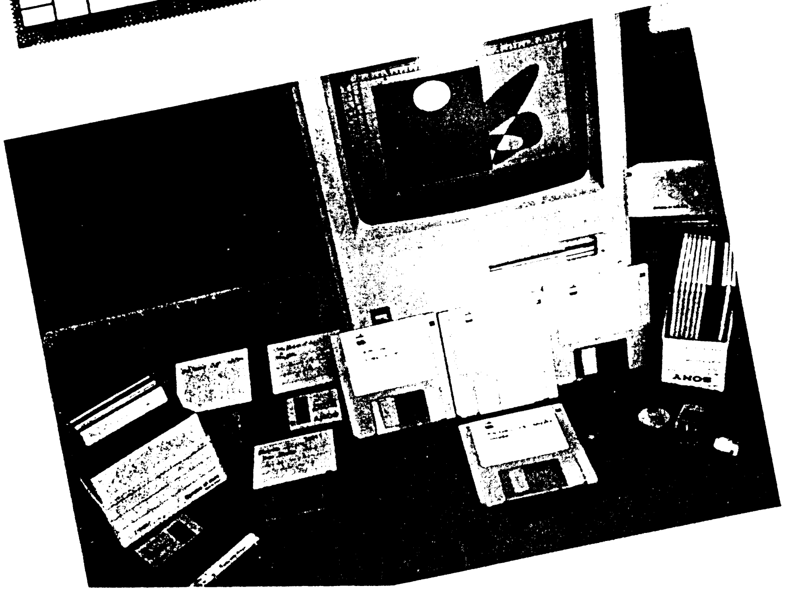
Mark Cutter, above left, created LisaDraw, the powerful graphics editor upon which MacDraw is based. Lisa product manager Randy Battat, above right, believes that Apple's unorthodox marketing strategy encourages the development of uniquely creative products.

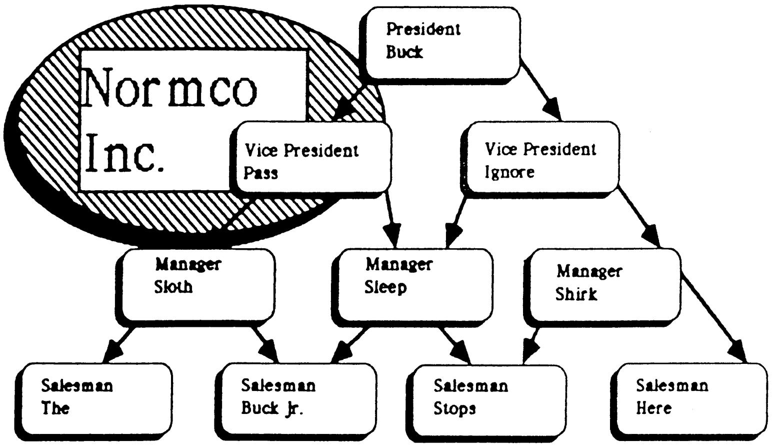


to a Draw



File
Edit
Type
Font
Layout
Arrange
Fill
Lines
Pen
Synergy
Text
Angle Of Trajectory
Energy
Text
Synergy . . .





The original directive to those creating the Draw software was to make a simple Org Chart Generator.

Deviation from the Norm

Battat begins the tale of Apple's unique method of product development by explaining the "normal," as opposed to Apple's, way of doing things. In the "normal" mode of doing things, innovative marketing people go out and survey the market. They ask people what they need, what they want, and what they would be able to use. Based on these surveys, they come back and tell the engineers what to build. Then, in that most perfect of theoretical worlds, the software engineers write programs to fit the niche the market has defined.

But we don't need Battat to tell us that Apple has a different style. Until the introduction of the IIc, Apple never designed a machine specifically targeted by a demand in the marketplace. Apple started out by building a computer that nobody needed *or* wanted. A personal computer? What on earth for? At least that's what most of us thought. When the Apple II turned out to be only slightly bigger than a typewriter and hundreds of times more useful, a need—for a personal computer—was created by its fill, the Apple II.

Similar stories could be told about the public's lack of desire for the III, the IIe, and most assuredly the thirty-two-bit projects. Who needs mice? Some critics, most notably Peter McWilliams, the author of several microcomputer books, have done a good job of pointing out Apple's inability to perceive the needs of the marketplace and to design a machine to meet those specific needs. But so what? Apple continues to build machines that no market surveys recommend, that nobody wants and nobody needs, but that everybody buys. So there's got to be something to this maverick company's unorthodox marketing strategy.

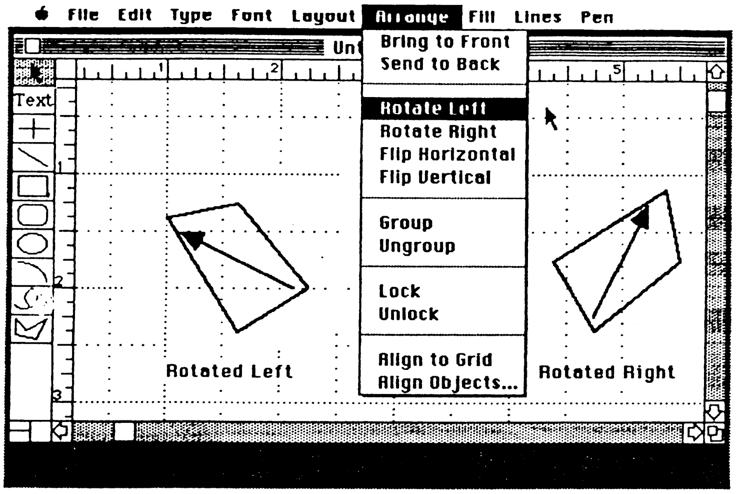
It seems that Apple used to have fairly complex, but conventional, ways of starting a project—strategies like New Product Proposals, Preliminary Marketing Requirements Documents (PMRD), Marketing Requirements Documents (MRD), and External Reference Specifications. Sounds boring, doesn't it? Battat says boring is an understatement. Anyway, a number of these documents had to be written, filed, and evaluated before a project could even get off the ground. Consequently, says Battat, "Many proposals never got off the drawing board." It's difficult to imagine anyone ever reading these documents, let alone getting excited over them.

This paper-shuffling on the *MacDraw* project was permanently done away with by Apple software engineer Mark Cutter, the author of *LisaDraw* and *MacDraw*. Cutter says that when he first started work on the project, paper-pushers hounded him day and night for things like PMRDs and External Reference Specifications. He says it's pretty difficult to write specifications before you even know exactly what you're trying to build. The papers could have been written, but Cutter knew it was either write the papers or write the code; he told Apple he didn't have time to do both.

At first, this didn't sit too well with management, but when people began passing mock-ups (simple working demonstrations of ideas for software) of *LisaDraw* around Apple, the paper-pushers stopped pushing. They thought Cutter's work was fantastic. According to Randy Battat, a good mock-up is a sure way to get the attention of the people with the resources at Apple. Indeed, he says, the filing cabinets are full of new product proposals that will never see the light of day simply because the engineers never mocked them up. So if you have an idea for a program, better mock it up before trying to get funding.

Cutter wasn't originally hired to work on the thirty-two-bit projects. He was supposed to be writing a graphics editor for the Apple III. But one day he wandered out to the Lisa building and started chatting with Bill Atkinson, who did all of the low-level graphics routines for the Lisa and the Mac and played a major role in the overall design of the Lisa, contributing to the design of the user interface and to some extent the design of the hardware as well.

Cutter and Atkinson spent three or four hours talking before Atkinson realized he wasn't talking to the new graphics person for the Lisa. Apple had been negotiating with someone to write the graphics editor for the Lisa, and



MacDraw is capable of rotating objects in ninety-degree increments only. Cutter would eventually like to see arbitrary rotation of shapes.

Atkinson thought Cutter was that person. But the person Apple was negotiating with never came aboard; so, just like the stand-in who gets to go on for the indisposed lead on opening night, Cutter soon got his wish to join the Lisa project.

(To be sure, this is only an isolated example, but perhaps you're wondering how such a successful organization as Apple can afford to go about getting its software engineers in such unorthodox ways. Randy Battat puts it this way: "Old theory," he says, "has a lot to do with structure. Structure is good," he pauses, searching for *something* good to say about structure, "...because you can measure it." Structure gives the paper-pushers something to measure, as in: "Oh, the External Reference Specifications call for an 'A' here and your program gives a 'B'; 'B' certainly works better, but you failed 'cause your spec says an 'A.'")

A Method to This Madness

Battat has worked at Apple for a number of years, and he has found that the project engineers who succeed all do certain things the same way. They stick to a methodology that has evolved over the years. And those who fail are those who ignore these methods.

According to Battat, the first thing that an Apple engineer with an idea for a product should do is "demonstrate an early complete prototype that doesn't aim to do much but that does it well." He says an idea that can be viewed on the screen is much more exciting and likely to get funded than an idea, no matter how good, on paper. Cutter, coincidentally, expressed a similar belief when describing why he decided to do a mock-up rather than a lot of paperwork: "Ideas are fine, but experience is much better."

Believe it or not, the first directive to those creating the *Draw* software was to make a simple Org Chart Generator. You know, an organizational chart—vice presidents are always drawing them up for other vice presidents so everyone will know where everyone stands in a given week. Although *LisaDraw* started out that way, it has evolved tremendously. Let's look at how it all started.

Mocking Up MacDraw

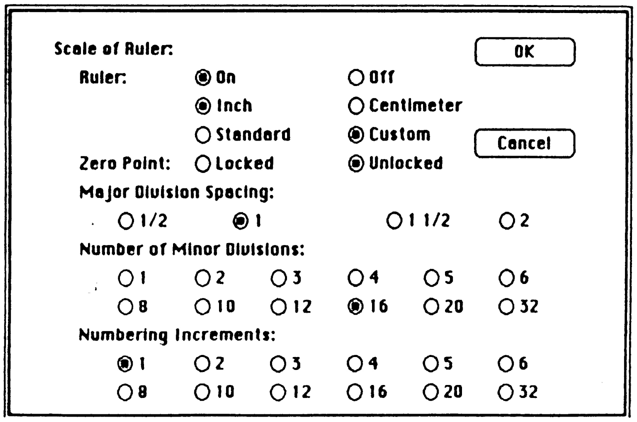
A very elaborate working prototype of a graphics editor had been built by Rod Perkins before Cutter came aboard. Cutter and Perkins spent a month or two meeting and discussing what the graphics editor should ultimately do, but they weren't able to come up with any satisfactory designs; nevertheless, they drafted a design from that first prototype. When Bill Atkinson saw it, he was decidedly unimpressed. So he went out himself and in a few days mocked up a completely different prototype.

Atkinson's prototype contained the basic ideas out of which *LisaDraw* and eventually *MacDraw* would evolve. It drew an object from a prototype palette and had handles one could use to stretch it. It was a very simple working version with only a few shapes and options. Once Atkinson had mocked it up, he went back to finish QuickDraw, the graphics primitives in the ROM, and Rod Perkins went on to write *LisaCalc*.

Well, the next few iterations of *LisaDraw* followed precisely Battat's first unwritten rule: "Get an early working mock-up." Cutter proceeded to get up several early working versions of *LisaDraw*. In fact, he got the next and successive versions up so fast that the usually overly optimistic marketing people had to keep revising the release date of the product *forward* in time. Originally scheduled for release six months after Lisa, *LisaDraw* was released at the same time as Lisa, despite the fact that it was started a year later than some of the other Lisa software.

The Power of Suggestion

Battat insists that his second observation about how software is most efficiently developed at Apple is just as important as the first.



Scale of Ruler:

Ruler: ● On ○ Off

● Inch ○ Centimeter

○ Standard ● Custom

Zero Point: ○ Locked ● Unlocked

Major Division Spacing:

○ 1/2 ● 1 ○ 1 1/2 ○ 2

Number of Minor Divisions:

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6

○ 8 ○ 10 ○ 12 ● 16 ○ 20 ○ 32

Numbering Increments:

● 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6

○ 8 ○ 10 ○ 12 ○ 16 ○ 20 ○ 32

A movable zero, or reference, point allows the user to show one ruler for one area or object in a drawing and then turn on a new ruler with a new reference point at another location.

There must be "a willingness, on the part of the software people, to let people play with the early versions of a product and a willingness [on the part of the authors] to listen to what they have to say." In other words, an engineer has to make duplicates of his product, pass them around Apple, and then maintain a constant openness to revision. He believes that this is the stage where creativity is embedded within the project, as a direct result of the harmonious and sharing environment at Apple. Without mentioning any names, Battat goes on to tell about products that had failed, he felt, because their authors had an attitude of, "Oh, I don't want to show it to you; it's buggy." Many good suggestions have been the result of releasing a product in-house.

The ability to lock down an object in *LisaDraw* and *MacDraw* was requested by a frustrated early user who accidentally kept selecting the wrong object when trying to manipulate very small graphics right next to each other. That lock-down feature, as well as the movable zero-point, were suggestions from Apple's own Wayne Rosing. Rosing managed the hardware development of the Lisa and also was technical manager of all of the engineers for the Lisa project. In Cutter's view, "Wayne is one of the greatest managers and executives because he would go home and *use* the software."

Rosing was adding a shop to his home and took an early version of *LisaDraw* to use in laying out the shop. It was while using the program to design the workshop that Rosing wished some of the objects could be locked down. This was also when he felt the need for a movable zero, or reference, point. A movable zero point allows the user to show one ruler for one area or object in a drawing and then turn on a new ruler with a new reference point at another location. It works like a tape measure. When you want to measure a wall, and then measure the length of a desk against that wall, you don't stretch the tape measure out, leave it, and then try to determine the length of the desk by subtracting it from the length of the wall. You simply bring the tape measure to the edge of the desk. Well, like a tape measure, *MacDraw*'s rulers can move too.

One suggestion for *MacDraw* came from a rather unusual source. It might have come from one of you; Cutter doesn't know the name of the person who made this suggestion, but he's grateful. Cutter had volunteered to answer questions at the West Coast Computer Faire for Apple. A patron who had been using *LisaDraw* for some time suggested, "Why not show the size of the arc when RESHAPE ARC is turned on with RESHAPE POLYGON?" Cutter thought that was a great idea and went home and implemented it immediately on *MacDraw*. (So next time you go to a computer show, be sure to ask whether the product you are trying out is under development. If it is, and you can make a worthwhile suggestion, you might just see your idea implemented.)

Cutter says he would like to implement "Undo," which is on *LisaDraw*, but he doesn't think he will have time to do that before *MacDraw* is released. He would also like to implement text rotation, but once again there might not be enough time before his deadline. These are both bells and whistles that might or might not make it in by *MacDraw*'s release, but at press time it wasn't known whether they would.

Cutter would also like to see arbitrary rotation of shapes. Right now *MacDraw* supports ROTATE RIGHT or ROTATE LEFT in ninety-degree increments only, but suppose somebody wanted to rotate just a few degrees. That capability would be fairly simple to implement for most shapes, but it would require new low-level graphics to work for ovals. It took Bill Atkinson several years to implement his low-level graphics routines. Anyone want to try to beat his time? Cutter would like to see 3-D implemented by someone else. "I could certainly do it, but I wouldn't be learning anything. I've written this graphics editor twice....I think I want to learn more than this." He is pretty much finished with the *Draw* programs; someone else has been assigned to update *LisaDraw*, and there will probably be another engineer on *MacDraw* updates.

The Last Few Bugs Are Hard To Find

The next step in the Apple product development cycle is what Randy Battat calls "gettin' it out the door." It's the step that many amateur programmers often ignore. If you've ever designed a program on your own, you may have gotten it just so bug-free and then left it. Commercial programs certainly shouldn't be released with *any* known bugs. Battat puts it mildly when he says that getting a product bug-free takes "a lot of blood, sweat, and tears and guts-crankin' late nights."

Debugging a program can be a frustrating and time-consuming task, and Cutter says that the last few bugs are always the hardest ones to find. A good example of almost not "gettin' it out the door" was the *MacPaint-MacWrite* disk. It seems that the final tweaks weren't applied to the master disk, from which thousands were to be duplicated, until just *two hours* before the actual duplicating began.

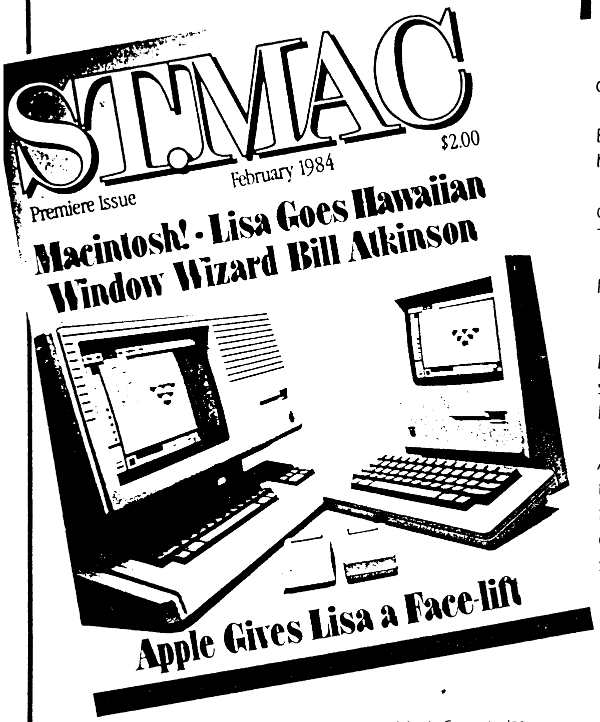
Battat believes that software engineers should set firm deadlines throughout the process of software development. People work better under a little bit of pressure. Programming is a creative process and therefore is subject to problems common to the arts: problems like the lack of inspiration. Can you imagine how dull *MacDraw* would be if Cutter had stayed with the original plan and had written an Org Chart Generator! And software arts is a competitive field. Dan Eilers, former product marketing manager for *LisaDraw* and now the treasurer of Apple Computer Inc., says "Always be nervous about competitors."

In the opinion of one West Coast Computer Faire attendee, "They'll buy anything that even boots on the Mac." This attitude of "write something quick and dirty and let's make a buck" comes from the fact that traditionally the largest sellers in software have always been the early birds, because there are so few available products at first. This creates fierce competition for companies like Apple that try to commit to quality software.

An analogy: Suppose someone was able to manufacture a plane that could fly at almost the speed of light. Now suppose it could be sold for the price of a Volkswagen. Finally, suppose there were only three airstrips in the world long enough to land such an airplane. A couple of things are going to happen. First, those airstrips are going to be pretty popular places. Second, the owners of the airstrips can charge whatever they please to people who land there; and third, you're going to have a mad rush of people going out and trying to lengthen their runways.

"Slipping the date" is what it's officially called when any of the many steps a product has to go through doesn't get finished in time. Given the above analogy, the Mac is the plane and the software is the runway the plane uses.

A lot of people out there are starting to extend the old, or build new, runways. Dan Eilers recently summed up his philosophy in the competitive software game by saying, "If I've got software, I'd say ship it." In the coming months, we'll see whether the Macintosh software development team did indeed "ship it."


ST.MAC
$2.00
February 1984
Premiere Issue
Macintosh! - Lisa Goes Hawaiian
Window Wizard Bill Atkinson
Apple Gives Lisa a Face-lift

There's Only One

There's only one Lisa Technology for personal computers. It's found only on Apple's Lisa and Macintosh computers.

There's lots of imitators. They have windows, or a mouse, or both. But they don't have Lisa Technology. Without Lisa Technology, you don't have state of the art. In state of the art, there's only one. It's Apple.

There's only one magazine that's made a monthly commitment to covering that technology. Eventually, there'll be many magazines. Lisa Technology is too exciting for journalists to ignore.

But in the beginning there's only one that'll exclusively cover Lisa and Macintosh on a regular, monthly basis.

You're reading it.

You can continue to read it for $24 a year. Unless you're a Lisa or Macintosh owner. Then you can get a free trial subscription by merely sending us your name, mailing address, and the serial number of your Mac or Lisa.

When you subscribe, you'll be getting a magazine published by the Apple experts. *Softalk* is the most complete Apple II and III magazine, so there's lots of insight to draw on for *ST.Mac*. You'll get news, reviews, features, user tips, thought-provoking analysis—all done in conversational, enjoyable English. Softalk Publishing wrote the book on easy-to-understand computer magazines. *ST.Mac* is in that tradition.

Subscribe today.

ST.Mac
Box 7041
North Hollywood, CA 91605

Apple, Lisa, and Macintosh are registered trademarks of Apple Computer Inc.
Softalk and ST.Mac are trademarks of Softalk Publishing Inc